Sure, I can provide a basic example of a recommendation system using collaborative filtering in Python. Collaborative filtering suggests items to users based on the preferences and behavior of other users. Here’s a simplified code example:import pandas as pd

From sklearn.metrics.pairwise import cosine\_similarity

# Sample user-item ratings data

Data = {

‘User’: [‘User1’, ‘User2’, ‘User3’, ‘User4’],

‘Item1’: [5, 4, 0, 0],

‘Item2’: [0, 5, 4, 0],

‘Item3’: [4, 0, 0, 5],

‘Item4’: [0, 0, 5, 4]

}

Df = pd.DataFrame(data)

# Calculate cosine similarity between users

Similarities = cosine\_similarity(df.iloc[:, 1:])

# Convert the similarity matrix into a DataFrame

User\_similarity\_matrix = pd.DataFrame(similarities, columns=df[‘User’], index=df[‘User’])

# Function to get recommendations for a user

Def get\_recommendations(user):

User\_ratings = df[df[‘User’] == user].iloc[:, 1:]

Similar\_users = user\_similarity\_matrix[user].sort\_values(ascending=False)[1:]

Recommendations = []

For item in user\_ratings.columns:

If user\_ratings[item].values[0] == 0:

Weighted\_sum = sum(user\_ratings[item] \* similar\_users)

Norm = sum(similar\_users)

Recommendations.append((item, weighted\_sum / norm))

Recommendations.sort(key=lambda x: x[1], reverse=True)

Return recommendations

# Example usage

User = ‘User1’

Recommendations = get\_recommendations(user)

Print(f”Recommendations for {user}:”)

For item, score in recommendations:

Print(f”{item}: {score}”)This code calculates the cosine similarity between users and provides recommendations for a given user based on their preferences and the preferences of similar users. You can replace the sample data with your own user-item ratings for movies, books, or products.